526,803

(12)特許協力条約に基づいて公開された国際出願

(19) 世界知的所有権機関
国際事務局

(43) 国際公開日
2004 年3 月25 日 (25.03.2004)

PCT

(10) 国際公開番号
WO 2004/025463 A1

(51) 国際特許分類[7]: G06F 9/44

(21) 国際出願番号: PCT/JP2003/011486

(22) 国際出願日: 2003 年9 月9 日 (09.09.2003)

(25) 国際出願の言語: 日本語

(26) 国際公開の言語: 日本語

(30) 優先権データ:
特願2002-262670 2002 年9 月9 日 (09.09.2002) JP
特願2003-10454 2003 年1 月17 日 (17.01.2003) JP

(71) 出願人 (米国を除く全ての指定国について): ソフトウェア生産技術研究所株式会社 (THE INSTITUTE OF COMPUTER BASED SOFTWARE METHODOLOGY AND TECHNOLOGY) [JP/JP]; 〒108-0074 東京都 港区高輪 三丁目１１番３号 Tokyo (JP).

(72) 発明者; および

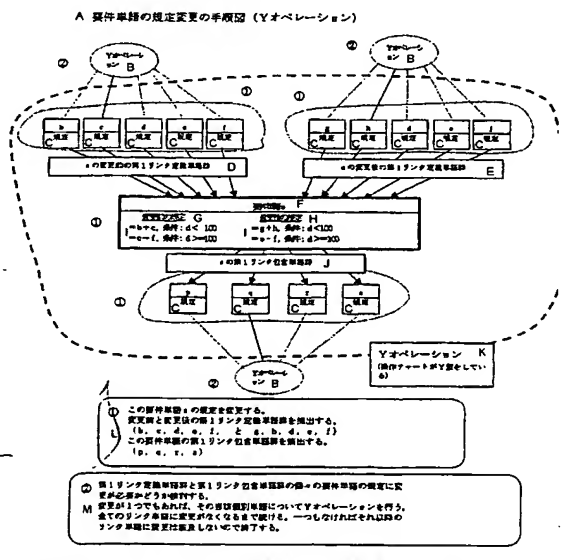
(75) 発明者/出願人 (米国についてのみ): 平山 貞宏 (HIRAYAMA,Sadahiro) [JP/JP]; 〒145-0066 東京都 大田区南雪谷 四丁目２２番１４号 Tokyo (JP). 根来 文生 (NEGORO,Fumio) [JP/JP]; 〒248-0001 神奈川県 鎌倉市十二所 967-64 Kanagawa (JP).

(74) 代理人: 友野 英三 (TOMONO,Eizo); 〒180-0023 東京都 武蔵野市境南町 二丁目２４番１０号 Tokyo (JP).

[続葉有]

(54) Title: REQUIREMENT DEFINING METHOD, METHOD FOR DEVELOPING SOFTWARE, METHOD FOR CHANGING REQUIREMENT WORD, AND NEWLY DEFINING METHOD

(54) 発明の名称: 要件定義方法、ソフトウェアの開発方法、及び、要件単語の変更方法並びに新規規定方法



A...CHART SHOWING PROCEDURES OF CHANGING DEFINITION OF REQUIREMENT WORD (Y OPERATION)
B...Y OPERATION
C...DEFINITION
D...FIRST LINK DEFINITION WORDS BEFORE CHANGE OF a
E...FIRST LINK DEFINITION WORDS AFTER CHANGE OF a
F...REQUIREMENT WORD A
G...DEFINITION BEFORE CHANGE
H...DEFINITION AFTER CHANGE
I...CONDITION
J...FIRST LINK INCLUSION WORDS OF a
K...Y OPERATION (THE OPERATION CHART IS SHAPED LIKE LETTER Y)
L...DEFINITION OF REQUIREMENT WORD a IS CHANGED
FIRST LINK DEFINITION WORDS BEFORE AND AFTER CHANGE ARE EXTRACTED.
(b, c, d, e, f and g, h, d, e, f)
FIRST LINK INCLUSION WORDS OF THE REQUIREMENT WORD ARE EXTRACTED.
(p, q, r, s)
M ...NECESSITY TO CHANGE EACH OF DEFINITIONS OF FIRST LINK DEFINITION WORDS AND FIRST LINK INCLUSION WORDS IS EXAMINED.
IF EVEN ONE CHANGE IS FOUND OUT, Y OPERATION ON THE CORRESPONDING WORD IS EXECUTED.
PROCESSING IS CONTINUED UNTIL NO CHANGE IS FOUND FOR ALL LINK WORDS. IF NONE, PROCESSING IS ENDED BECAUSE CHANGE DOES NOT INFLUENCE THE OTHER LINK WORDS

(57) Abstract: An output item finally determined by computer software to be developed is extracted, and the output data is defined by using a data generation formula using intermediate data items. The definition processing of defining all the intermediate data items by using different data generation formulae is repeated until generation of a data item is due to input data. The definition including the thus generated data items and data generation formula execution conditions is used as a requirement definition. The thus-made requirement definition and the processing order of data items are automatically found out. Alternatively data is automatically rearranged. As a result, a program is automatically created. Such a method is applied to development of software. When requirement words are changed, words that are different from the requirement words before and after the change and define the methods of generating the requirement words are extracted and presented. Further, out of the different words, words the generation methods of which are defined by the requirement words are also extracted and presented. The necessity of the word change is examined. If the change is necessary, the change processing is continued; if not, the change processing of the requirement words is ended.

[続葉有]

WO 2004/025463 A1

(81) 指定国 (国内): AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, BZ, CA, CH, CN, CO, CR, CU, CZ, DE, DK, DM, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, MZ, NI, NO, NZ, OM, PG, PH, PL, PT, RO, RU, SC, SD, SE, SG, SK, SL, SY, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, YU, ZA, ZM, ZW.

(84) 指定国 (広域): ARIPO 特許 (GH, GM, KE, LS, MW, MZ, SD, SL, SZ, TZ, UG, ZM, ZW), ユーラシア特許 (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), ヨーロッパ特許 (AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HU, IE, IT, LU, MC, NL, PT, RO, SE, SI, SK, TR), OAPI 特許 (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, ML, MR, NE, SN, TD, TG).

添付公開書類:
— 国際調査報告書
— 請求の範囲の補正の期限前の公開であり、補正書受領の際には再公開される。

２文字コード及び他の略語については、定期発行される各*PCT*ガゼットの巻頭に掲載されている「コードと略語のガイダンスノート」を参照。

(57) 要約:

開発対象のコンピュータソフトウェアが最終的に求めるアウトプット項目を取出し、このアウトプットデータを中間データ項目を用いたデータ生成式により規定する。そして、これらの中間データ項目の総てを、それぞれ別のデータ生成式により規定する処理をデータ項目の生成がインプットデータによることになるまで繰返す。その結果、得られたデータ項目及びそのデータ生成式実行条件等の規定を要件定義とする。さらに、得られた要件定義と、データ項目の処理順序を自動的に見出して、或いはデータを自動的に正しい順序で成立させてプログラムを自動生成する方法を適用し、ソフトウェア開発を行う。また、要件単語群を変更する際に、変更前と変更後の該単語の生成方法を定義する別の単語及び他の単語のうち該単語によって生成方法が定義される単語とその定義を摘出して提示し、変更の要否を検討し、「要」なら変更を継続し「否」なら該単語の変更を終了する。

明　細　書

要件定義方法、ソフトウェアの開発方法、及び、要件単語の変更方法並びに新規規定方法

技術分野

本発明は、ソフトウェア開発手法に係り、特に、連鎖式データ項目規定法による要件定義方法、コンピュータソフトウェアの自動開発方法、及び連鎖式データ項目規定法による要件定義法についての要件単語の変更方法並びに要件単語の新規規定方法に関する。

背景技術

従来のソフトウェア開発手法によれば、基本的に総てのプログラムを、（１）順序処理、（２）選択分岐処理、（３）反復処理の３種類の基本命令文の組み合わせだけで作る。ここで、（１）順序処理とは、記述されたプログラム文の順にデータ処理を実行する処理をいい、（２）選択分岐処理とは、条件を判断してデータ処理の実行順序を、記述されたプログラム文の順序から他のプログラム文へ分岐させる処理をいい、（３）反復処理とは、ある条件が成立している間は記述されたプログラム文を繰り返し実行する処理をいう。

これらの命令文は総て処理順序を指示するものであり、総てのプログラムはこれらによってデータ項目の処理順序を人が総て指示しなければならない。データ項目が多かったり実行条件が複雑多岐である錯綜する業務システムでは、プログラミングによって正しい処理順序を指示するためには、人は業務システムをよく理解し総てのデータ項目の処理順序

を正しく把握しなければならない。この点、一部のデータ項目だけの処理順序を把握しても、処理順序は全体のデータの関係を決めなければならないので意味がない。

このように正しい処理順序でプログラミングすることを最終目的としつつ、そのための準備作業として業務分析、業務デザイン、システムデザイン、プログラムデザイン等が行われ、そこでは例えば下記のようないろいろな工夫がなされている。

・業務システムの中でのデータの流れを示す図としてデータフローダイアグラムを作成する。

・データ処理の要素機能（例えば販売金額を「単価×数量」として計算する機能）を幾つか含んだデータ処理機能のあるまとまり（例えば個々の販売金額を計算し更にそれらを合計する機能）を正しい処理順序で作り（「モジュール」と呼ぶ）、更にそれらのモジュールを正しい順序で組み合わせることによって全体のプログラムを作る。

・データ処理の順序を示すフローチャートを作る。

・データ処理の途中で中間データ項目を作って考えを整理する。

・データ処理の途中で中間ファイルを作って考える範囲を限定する。

従来のソフトウェアの開発方法は、業務システム側から見れば、業務システムの個々の要素機能（例えば販売金額を「単価×数量」として計算する機能）を決め、次にそれらをどんな順序で実行するかを決めてそれらをプログラムでコンピュータに指示するというものである。前述の通りコンピュータは処理順序を自ら探すことはできないので総て人が指示しなければならない。業務の要素機能は、ソフトウェアのユーザである業務部門が決めることであるからユーザーズ・ロジック（Users' Logic）と呼び、それらの処理の実行順序をコントロール・ロジック（Control Logic）と呼ぶ。

従来のプログラミングとは、端的に言えば、与えられたユーザーズ・ロジック（Users' Logic ）と正しいコントロール・ロジック（Control Logic）とをコンピュータに指示することである。個々のユーザーズ・ロジック（Users' Logic ）はユーザが必ず決めなければならないものであるが、これは比較的簡単に決めることができる。プログラミングの作業が膨大になる理由はコントロール・ロジック（Control Logic）を正しく指示することが容易ではないことに求められる。そのための準備や工夫として、先に述べた詳細な業務分析、業務デザイン、データフローダイアグラムの作成、プログラムデザイン、モジュールの作成、フローチャートの作成、中間データ項目の設定、中間ファイルの設定等が行われている。これらは総て正しい順序でプログラムを書くための準備作業である。それらの後にプログラミングによって正しい処理順序のコントロール・ロジック（Control Logic）を指示する。

以上をまとめれば、従来、データ処理順序は人間がプログラムによってコンピュータに指示しなければならなかった一方、錯綜したシステムでは、総てのデータ項目の処理順序を人間がコンピュータに間違いなく指示することは容易なことではない。このため全体システムの総てのデータの流れを正しく理解把握しなければならなくなり、業務分析、業務デザイン、データフローダイアグラムの作成、プログラムデザイン、等々のプログラミングの前の準備やプログラミングの膨大な作業が必要となってしまう。そのため、コストも時間も膨大なものとなってしまっているのが現状である。

上記の問題をさらに詳細に検討する。

従来のソフトウェア開発方法は、前述の通りコンピュータがデータ項目の処理順序を自ら見出すことができないので人間が総て指示しなければならないが、すこし複雑なシステムでは処理順序は複雑に錯綜しそれ

らの全体を理解把握することは容易ではない。このことが原因となり下記のような問題が起きている。

・ソフトウェア開発に膨大な手間、時間、コストが掛かる。

・エンドユーザが業務システムをデザインするためにはある程度のシステムエンジニアリングの知識が必要であることから、結局、システムエンジニアに過度に依存し勝ちになる。

・実務経験がなく業務の運営・成果責任はないシステムエンジニアがリーダーシップを取り勝ちになる。

これらの問題点に加えて、システムエンジニアが主導するので、エンドユーザが本当に望むソフトウェアができるまでにやり直しが多くなる、必要なタイミングを逸することも多い、といった問題点もあった。

一方、プログラムの変更は常に必要である。新規に開発するプログラムの場合、初めに業務システムやプログラムシステムが総て決まっていることはむしろ少ないので、プログラムはプログラミング過程で変更しながら作成して行かなければならないことは通常の現象である。また既に完成されたプログラムでも、業務環境や経営の方針の変化に応じて要件は常に変化しているので、それに応ずるプログラムの変更は常に必要である。

しかしながら従来、新規開発の途上にあるプログラムにおいても、或いは既に一度は完成された既存のプログラムにおいても、プログラムの変更は容易ではない。その根本原因は従来法では処理順序を人がプログラムによってコンピュータに指示しなければならないことである。このため、要件の変更に応ずるプログラムの変更が非常に厄介であり、要件の変更には膨大な人力、コスト、時間がかかる、という問題があった。その事情をもう少し詳しく説明する。

上述したように、従来法のプログラムではその命令文の順序にしたが

ってデータ処理が実行される。したがって処理順序は人が命令文を書く順序によって或いは条件付分岐の行く先を書くことによって指示しているし指示しなければならない。このように処理順序が決められているので１つのプログラムの中の１つのデータ項目の値はそれが処理されるまでの前処理と当該の処理によって決まる。したがって１つのデータ項目の値は１つのプログラムの中の何処に書かれているかによって変わってくる。したがって前処理を理解してさえいれば１つのデータ項目は１つのプログラムの中で別のデータ生成式を持ってもよいし別の値を持ってもよい。つまり「１つのデータ項目は１つのプログラムの中でどんな前処理をされたかに拘わらず同じ値を持つ」ということではないのである。結局、処理順序を人が決めてやらなければならない代わりにデータ項目の定義や値に自由度が出てきてしまうのである。だから１つのデータ項目について１つのプログラムの中で何処でどんな定義で使われているのかはプログラムを作った人以外にはわからない。作った人でも長くは覚えてはいない。

例えばプログラムの中の一部に

y=ax+b

y=y+1

y=y+1

y=y+1

という命令文があったとする。最後のｙは最初のｙに３を足したものとなる。即ちｙは処理段階の何処にいるかによって異なる値を持ち得るのである。この従来法の性質はＣＯＢＯＬ，C++，Java（登録商標）等のプログラム言語の種類に拘わらず結局は同じである。

さて、この時 y=ax+b をある段階から y=cz-d に変更したいとする。プログラムの中ではｙはいろいろな処理方法を指示されていろいろのと

ころに潜んでいる。これらを正しく変更しなければならない。しかしながら困ることは、

① プログラムは人が法則もなく勝手に作ることができるので、y がプログラム全体の何処にあるか分らない。

② y の前処理を遡ってたどって理解しなければ y のデータ生成式をどう変更してよいか分らない。

③ y は ax+b という形で表現されているかもしれないので y だけを探しても十分とは言えない。

④ y を変更することによりその波及効果として他のプログラム（例えば x や z のデータ生成式）も変更しなければならないかもしれない。

⑤ また条件を付加して選択分岐を行わせるという変更の場合、その変更に応じてプログラムの他の部分においても条件付選択分岐の形に変更しなければならない。

等であるがそれらの総てを見つけて正しく変更することは容易ではない。

結局従来法においてプログラムの変更作業が困難である理由は、1箇所の変更に応じて他に変更すべきものが膨大なプログラム命令文の何処に潜んでいるのか容易に分らないことである。その根本原因は，繰り返しになるが、

・要件をコンピュータが理解できるように「人」が個々の業務要素の処理方法(Users' logic) の処理順序(Control logic)をプログラムの形にして指示しなければならないこと、したがって

・プログラムは「人」が勝手に処理順序(Control logic)を考えて作ったものであるからプログラム全体の中に何が何処にどんな形で書かれているのか分らない。1箇所変えればそのことによるプログラム全体への影響も機械的には分らない、

ことである。

従来法のプログラムは変更について以上のような困難性を内在している。したがって、総てを正しく変更するためには、人がプログラムの中の総ての命令文を詳細にたどって変更すべきところを摘出して変更することしかないのである。勿論業務の性質からプログラムのこの部分には変更が及ばないはずであるとして詳細なレビューをしないこともできるが、理論的には必要な変更はプログラムのどこにでも及ぶということを認識しなければならない。

以上に述べたのは、要件の変更に応ずるプログラムの変更についての従来法の特徴であった。しかしそもそもプログラムを新規に作成するときの従来法の特徴についても述べなければならない。

従来法では、前述の通り、データ項目の処理順序は人が逐一プログラムによってコンピュータに指示しなければならなかった。しかし、錯綜したシステムでは、総てのデータ項目の処理順序を人がプログラムによってコンピュータに間違いなく指示することは容易なことではない。処理順序を正しく指示するためには当然全体システムの総てのデータとその流れを正しく理解して把握しなければならない。そのために、業務分析、業務デザイン、データフローダイアグラムの作成、プログラムデザイン、等々のプログラミングの前の準備が必要でありまたプログラミングの膨大な作業が必要となる。このためコストも時間も膨大に掛かるのである。

もし要件(Users' logic)即ち要件単語の定義から特定の基本法則にしたがって処理順序(Control logic)を人が指示しなくともそれが自動的に導き出されることにより自動的にプログラムを生成することができるなら、要件の変更によるプログラムの変更は、要件単語の変更が正しく行われる限り、自動的に行うことができる。そうすればプログラムの変

更は極めて容易になる。この場合１つのプログラムの中の１つのデータ項目は１つの値しか持たないようにする。そうしなければそもそも自動プログラミングはできないであろう。連鎖式要件定義法と要件の変更方法は正にその特徴を持っている。

図３にその関係を図解する。図３についての詳細な説明は図をもってかえることとし、ここでは割愛する。

本発明は、このような従来の技術が有していた諸問題点を解決しようとするものであって、機械的に要件定義を行うことが可能な要件定義方法を提供することを目的とするものである。

本発明の更なる目的は、機械的に行われた要件定義とそれに基づいてデータ項目の処理順序を自動的に見出して、或いはデータを自動的に正しい順序で成立させて行って、プログラムを自動的に開発する手法（例えば「Ｌｙｅｅ（登録商標）」）を用いることを適合させることによりソフトウェアの開発を自動的に行うというコンピュータソフトウェアの自動開発方法を提供することにある。

ここで「Ｌｙｅｅ（登録商標）」とは、本願発明に係る発明者の一人である根来文生により発明されたソフトウェア生産方法等に係る発明、技術のことをいい、その詳細は例えば、国際公開ＷＯ９７／１６７８４　Ａ１パンフレット（以下、「特許文献１」ともいう。）、国際公開ＷＯ９８／１９２３２　Ａ１パンフレット（以下、「特許文献２」ともいう。）、国際公開ＷＯ９９／４９３８７　Ａ１パンフレット（以下、「特許文献３」ともいう。）、国際公開ＷＯ００／７９３８５　Ａ１パンフレット（以下、「特許文献４」ともいう。）、国際公開ＷＯ０２／４２９０４　Ａ１パンフレット（以下、「特許文献５」ともいう。）、特開２００２－２０２８３号公報（以下、「特許文献６」ともいう。）等によって規定される諸国際公開公報等に開示されている。

本発明はまた、従来法のように先ず要件を総て定義し次にプログラミングを行うのではなく、要件の定義を初めから構築する作業それ自身を個々の要件単語の定義のみによって簡単にかつ効率的に行い、またその要件の変更に応ずる要件単語の規定の変更を簡単にかつ効率的に行うことが可能な、連鎖式データ項目規定法による要件定義法についての要件単語の変更方法及び要件単語の新規規定方法を提供することを目的とするものである。

また、新規に規定された要件単語群或いは変更された後の要件単語群を、単語の処理順序を問わないで自動的にプログラミングすることができる手法に適用すれば、人は従来法のような前述の膨大な作業をする必要がなくなる、という効果が生まれる。

発明の開示

本発明は上記目的を達成するために、本発明の第１課題解決手段は、（ａ）開発対象のコンピュータソフトウェアが最終的に求めようとするアウトプットデータ項目を総て取り出すステップと、（ｂ）該取り出されたアウトプットデータ項目の一をデータ生成式及びそのデータ生成式実行条件により規定するステップと、（ｃ）当該データ生成式及びそのデータ生成式実行条件を規定するために現れた新たなデータ項目の総てに対し、当該総ての新たなデータ項目のそれぞれを別のデータ生成式及びそのデータ生成式実行条件により規定するステップと、（ｄ）ステップ（ｃ）を、当該データ生成式を構成するのがインプットデータ項目のみとなるまで繰り返すステップと、（ｅ）ステップ（ａ）乃至ステップ（ｄ）を、該最終的に求めようとするアウトプットデータ項目の総てについて実行し、この実行の結果得られたデータ生成式及びそのデータ生成式実行条件、及び該当する場合にはそれらに加えてさらに、インプット／アウト

プットの属性区分及びそのデータ項目の存在する記録媒体の明示による規定を要件定義とするステップとを備えることとしたものである。

また、本発明の第２の課題解決手段は、（ａ）第１の課題解決手段に規定される要件定義方法を用いて要件定義を得るステップと、（ｂ）その結果得られた要件定義中に規定されたデータ項目に基づいて、データ項目の処理順序を自動的に見出すことで或いはデータを自動的に正しい順序で成立させて行くことでプログラムを自動的に開発する手法に適用し、該適用の結果所望のソフトウェアを得るステップとを備えることとしたものである。

上記第１の課題解決手段による作用は次の通りである。すなわち、本発明の連鎖式データ項目規定による要件定義法によって、目的であるデータ項目が決まりさえすればその規定から出発して簡単に機械的に要件定義を行うことができる。この点従来法では、この方法による要件定義はデータ項目の処理順序を示していないので役立たない。

また、上記第２の課題解決手段による作用は、本願発明の第１課題解決手段に係る機械的に要件定義を得、この機械的になされた要件定義に基づいて、データ項目の処理順序を自動的に見出し或いはデータを自動的に正しい順序で成立させて行って、プログラムを自動的に開発する手法（例えば「Ｌｙｅｅ（登録商標）」）を用いるので、結果的にでき上がるソフトウェアの開発は自動的とすることが可能となる。

本願発明の第１課題解決手段及び第２課題解決手段を用いる結果、データ処理順序を人間が考えプログラミングとしてコンピュータに指示するということはしなくてよくなるので、従来法で必要であった詳細な業務分析、業務デザイン、データフローダイアグラムの作成、プログラムデザイン、モジュールの作成、フローチャートの作成、中間データ項目の設定、中間ファイルの設定、さらにはプログラミング自体も人間が行

う必要がなくなる。従って、従来必要であった膨大なソフトウェア開発作業が大幅に縮減できる。これにより、開発時間とコストが大幅に削減できる。

因みに、従来から,データ項目群をそれを構成する個々のデータ項目及びそれと他のデータ項目との関係を示すポインタによって表現したリスト構造というデータ項目群の概念があり、それをベースにしたプログラミング言語がある。これに対して本願発明の第１課題解決手段は、データ項目群を創出するときに個々のデータ項目をデータ項目の規定だけで連鎖的に摘出する方法に係る。そこには上述のポインタの概念はない。また本願発明の第２課題解決手段は、第１課題解決手段で規定されたデータ項目群に基づき（リスト構造とは関係なくポインタなしで）データ項目の処理順序を自動的に見出して、或いはデータを自動的に正しい順序で成立させて行って、プログラムを自動的に開発する方法を用いることによりソフトウェアを自動的に開発する方法である。

また本発明の第３課題解決手段は、ソフトウェアの要件として規定される要件単語（＝データ項目）群に係る、該単語の名前、該単語に対応する値を得るデータ生成式（インプットによって値が得られることを含む。）、該単語に対応する値が成立するための条件（データ生成式実行条件）、該単語がインプットかアウトプットかの属性区分、該単語の存在する記録媒体で規定される情報を、規定された要件単語の配列の順序を問わないで自動プログラミングができる手法に適用して自動的に作成されるプログラムの要件の変更にあたり、（ａ）変更すべき要件単語自身の規定を変更（変更は削除、追加を含む。）する操作と、（ｂ）前記（ａ）の当該単語の規定の変更の影響によって規定変更が必要となる可能性がある要件単語として、当該要件単語の変更前と変更後の第１リンク定義単語と第１リンク包含単語とを摘出する操作と、（ｃ）前記摘出された個々

の単語について、その規定の変更が必要かどうかを検討する操作と、（ｄ）変更の必要がある単語について前記（ａ）乃至（ｃ）の操作を繰り返す操作とを備えることを特徴とする。

ここで、或る単語の「第１リンク定義単語」とは、その単語の規定に含まれる他の総ての単語、即ち、その単語を生成するために必要な他の総ての単語のことをいう。

また、或る単語の「第１リンク包含単語」とは、その単語が規定に含まれる他の総ての単語、即ち、その単語によって生成せられる他の総ての単語のことをいう。

さらに本発明の第４課題解決手段は、ソフトウェアの要件として規定される要件単語（＝データ項目）群に係る、該単語の名前、該単語に対応する値を得るデータ生成式（インプットによって値が得られることを含む。）、該単語に対応する値が成立するための条件（データ生成式実行条件）、該単語がインプットかアウトプットかの属性区分、該単語の存在する記録媒体で規定される情報を、規定された要件単語の配列の順序を問わないで自動プログラミングができる手法に適用して自動的に作成されるプログラムの新規開発においては、新規要件規定作業とその修正作業との両方が必要であるので、プログラムの新規開発にあたり、新規の要件規定は何もない状態からの変更と考えて、（ａ）変更すべき要件単語自身の規定を変更（変更は削除、追加を含む。）する操作と、（ｂ）前記（ａ）の当該単語の規定の変更の影響によって規定変更が必要となる可能性がある要件単語として、当該要件単語の変更前と変更後の第１リンク定義単語と第１リンク包含単語とを摘出する操作と、（ｃ）前記摘出された個々の単語について、その規定の変更が必要かどうかを検討する操作と、（ｄ）変更の必要がある単語について前記（ａ）乃至（ｃ）の操作を繰り返す操作とを備えることを特徴とする。

上記で規定される本願の第３または第４課題解決手段に係る発明では、要件変更に応じてプログラムを変更（ここでは新規に開発するときを含めて考える）するにあたり、データ項目の定義や値についての自由度が出てくる余地を持つ従来法ではプログラムの中の総ての命令文を詳細にたどって変更すべきところを、機械的に摘出して変更するので、結局は総てを正しく変更することが可能となる。

またさらに、本願は、その本質上、上記で規定される要件単語に係る諸情報のみを、規定された単語（データ項目）の配列の順序を問わないで自動プログラミングができる手法に適用することで、プログラムを自動的に作成するプログラムの開発方法、或いは、プログラムの改変方法、プログラムの保守方法として捉えることも可能である。

図面の簡単な説明

第１図は、要件単語の規定を変更（追加を含む）するときに、１つの変更の影響によって派生する必要な変更も同時に行いながら変更して行く手順を示している概念図である。

第２図は、データ項目間の関係を模式的に示すと共に、連鎖式要件定義法（Data Chain Requirement Definition (DCRD)法）を概念的に説明するための概念図である。

第３図は、要件定義と目的プログラムとの関係について、従来技術と本願に係る発明とを比較論的に説明するための概念図である。

発明の概要

連鎖式要件定義法は、ソフトウェアの要件を要件単語（＝データ項目）群の規定のみで定義する方法である。夫々の要件単語は、

単語の名前、

単語の値を得るデータ生成式（インプットによって値が得られることを含む。）、

単語の値が成立するための条件（データ生成式実行条件）、

単語がインプットかアウトプットかの属性区分、

単語の存在する記録媒体

だけを規定すればよい。

連鎖式要件定義は、意図を表すデータ項目から出発してまずその意図を導き出すデータ項目を規定し新たに現れたデータ項目を更に規定する。これを繰り返して次々に必要なデータ項目を引きずり出し、引きずり出されたデータ項目が総てインプットデータになるまで規定を続ける。この方法によって総てのデータ項目を引きずり出して規定する。それを要件定義とする。

従来法では、上記「従来の技術」の項で述べたように、データ項目の処理順序は人がプログラムによって逐一コンピュータに指示しなければならなかった。そのため全体システムの総てのデータとその流れを正しく理解して把握しなければならない。そのためにプログラミングの前に、業務分析、業務デザイン、データフローダイアグラムの作成、プログラムデザイン、等々の準備が必要でありまたプログラミングの膨大な作業が必要となる。このためコストも時間も膨大に掛かるのである。

これに対して連鎖式要件定義法では、データ項目の処理順序とは関係なくデータ項目を逐次的に規定して要件定義とするので、従来法において全部のデータ項目を初めに総て把握しそれらの処理順序を決めるための上記の厄介な準備作業が必要なくなり、非常に高い効率で要件定義ができるのである。

連鎖式要件定義法においてこのようにして規定された個々のデータ項目については、1つのプログラムの中で一義的に規定されているので1

つのデータ項目は１つのプログラムの中の各所で使われても１つの値しか持たないことが保証されている。連鎖式要件定義法のこの特徴は従来法のプログラミングと根本的に異なる点である。上記「従来の技術」の項の例では連鎖式要件定義法では

$$y=ax+b$$

$$y_1=y+1$$

$$y_2=y_1+1$$

$$y_3=y_2+1$$

として別々のデータ項目として規定するのである。

このようにして引きずり出され規定されたデータ項目の順番は、データ項目の処理順序とは全く異なるので、処理順序(Control logic)を人が指示しなければならない従来のプログラム開発手法では、このような要件定義は直接的には有効でない。しかし、処理順序不問の手法（例えばＬｙｅｅ（登録商標)。特許文献１乃至６参照）を用いればこの要件定義でも直接自動プログラミングができるので連鎖式要件定義法は完全に有効になり、要件定義法として連鎖式の高効率が生きることになる。

この方法において要件の変更とは上述の要件単語の幾つかについてその規定を変更することであるがそれはどのようにすればよいのであろうか。

一つ一つの要件単語を正しく変更するためには、個々の要件単語（例えば要件単語ａ）の規定の変更をするのみならず、その変更が他の要件単語の規定に与える影響についても検討しながら必要な変更をしなければならない。従来法ではそれが最も難しいのである。連鎖式要件定義法においてその変更方法を例を用いながら説明する。

変更すべき要件単語を単語ａとする。変更前の単語ａの定義としては、

$$a=b+c$$

成立条件：d<100

a=e-f

成立条件：d>=100

であるとする。

単語aの定義をまず変更し、次にその影響によって変更の可能性がある単語を摘出し、それらの個々の単語についてその定義を変更する必要があるかどうかを検討し、必要があれば変更する。その手順は次の通りである。図1（要件単語の規定変更の手順図）を参照されたい。

（a）先ず、変更する要件単語a自身の規定を変更する。（変更は削除・追加を含む。）

例えば a=g+h に変更したい場合或いは成立条件を変更したい場合、変更前のデータ生成式やデータ生成式実行条件その他のこの要件変更に応じて必要な要件単語規定を変更する。

削除の場合にも変更前の要件単語の定義が変更（削除）されることである。

新たに追加される要件単語の場合には変更前の要件単語の変更はなく新たに規定するのみである。

（b）次に、（a）の要件単語aの規定の変更の影響によって規定変更が必要となる可能性がある要件単語として、要件単語aの変更前と変更後の第1リンク定義単語群（下記＊1参照）及び単語リストの中の第1リンク包含単語群（下記＊2参照）を摘出する。

＊1 ある要件単語aの第1リンク定義単語とは、その要件単語aの規定に含まれる他のすべての単語である。その単語aを生成するために必要な他の総ての単語である。例えば要件単語aの第1リンク定義単語とは上記のb,c,d,e,fの単語である。

＊２　ある要件単語ａの第１リンク包含単語とは、その要件単語ａが含まれる規定をもつ他のすべての単語である。その単語ａによって生成される他の総ての単語である。例えば要件単語ａの第１リンク包含単語とは

ｐ＝ａ＊ｄ　とか

ｑ＝ｇ＋ａ

或いは

ｒ＝ｆ／ｈ　成立条件：ａ＜ｃ

等の場合におけるａを含むｐ，ｑ，ｒ単語である。

（ｃ）上記（ｂ）で摘出された個々の単語について、その規定の変更が必要かどうかを検討する。

（ｄ）上記（ｃ）の結果、変更の必要がある単語について上記の（ａ）、（ｂ）、（ｃ）と全くの同様の操作（Ｙオペレーション…図１参照）を繰り返す。

（ｅ）上記（ｄ）の結果、ある要件単語ａから派生する総ての単語の規定に変更の必要がなくなればある要件単語ａの変更操作は完了する。

ある要件単語ａに変更がなければ、その単語の第１リンク定義単語群にも第１リンク包含単語群にも変更はなく、要件単語ａが原因となる波及的変更をする必要がない。そこで変更の影響は遮断される。

何故変更の影響がその単語で遮断されるかというと、元々連鎖式要件単語の定義は１つの業務要素の規定を基にして次の業務要素を引きずり出す方法であるから、引きずり出す要素業務が変更前と同じであり変更がなければ、それ以降にその変更がなかった業務要素によって引きずり出される要素業務は変更前と同じになるのは自明だからである。もし引きずり出された業務要素に変更が必要ならそれは別件の新たな変更であ

り波及効果ではない。一方またある単語ａが他の単語の定義の中に使われていた場合、その単語ａに変更がないときには、他の単語も単語ａの波及効果としての変更はないことも自明である。したがって１つの要件単語の変更の影響は、変更がなくなった要件単語で遮断される。

以上の通り、本発明の連鎖式要件定義法における要件の変更方法は、１つの変更の影響が及ぶ範囲を限定でき、また変更の可能性のあるのはどんな要件単語であるかを提示する。これは総ての要件単語が連鎖式に生成されたことによって、互いに整合性を持った単語のネットワークを形成しており、したがって１つの単語は１つの値しか持たないことが保証されているからである。これは従来法のプログラムの変更とは根本的に違う点であり、遥かに優れているポイントである。この方法と処理の順序性を問わない自動プログラミング方法とがあいまって、ソフトウェアの開発・保守方法に劇的変化をもたらすものである。要件変更とプログラム変更が厄介な従来法の問題を根本的に解決するものである。従来法は前述の通り、１つのプログラムの変更に応じて他の部分で変更すべきプログラムがどこにあるのか、また、どの様に変更すべきかは人が考えなければならない。人が勝手に法則もなくプログラミングしたのでそれを見出すことは大変厄介なことである。理屈の上では変更の可能性はプログラム全体の何処にでもあるのでプログラム全体をしらみつぶしに探して考えなければならないのである。

上記の「発明の概要」の項で述べたことは、要件の変更方法であるが、要件を新規に定義するとき、変更すべき要件単語はない。しかし新規に要件単語を規定することは、何もない状態からの追加としての変更であるから、上記「発明の概要」の項で述べたことは新規の要件定義の方法でもある。したがって、要件単語の新規規定の場合にも、上記の（ａ）、（ｂ）、（ｃ）、（ｄ）、（ｅ）を行えばよいのである。

発明の最良の実施形態

（第１の実施形態）

本願に係る第１の発明（以下、「本願第１発明」ということもある。）について、簡単な例を用いて説明する。例えば販売金額を求めるソフトウェアを開発しようとするとき、先ず求めるアウトプットデータ項目である「販売金額」というデータ項目（単語）を下記の式のように規定する。次にそのデータ生成式に現れた新しいデータ項目（「販売単価」と「販売数量」）を夫々に規定する。同様のことを各データ項目がインプットデータと規定されるまで続け総てインプットデータとなれば完了する。

（１）　販売金額＝販売単価×販売数量、

（２）　販売単価＝仕入れ単価×（１＋マージン）、

（３）　販売数量＝インプットデータ、

（４）　仕入れ単価＝インプットデータ、

（５）　マージン＝インプットデータ、

因みにそれらの正しい処理順序は、個々のデータ項目の値が得られるためにはその前にそれに必要なデータ項目の値が得られていなければならないから、正しい処理順序について人は（３）、（４）、（５）、（この３つについては総てインプットデータ項目であるからそれら間の処理順序は問わない）、（２）、（１）の順であることを探り当てることができる。データ項目が多く、且つ、条件分岐も多い複雑なシステムでも、時間がどれほど掛かってもよければ、人は正しい順序を探り当てることは可能である。しかし残念ながら現在のソフトウェア開発手法では、コンピュータは処理順序を探り当てる機能を持っていない。「従来の技術」項に述べたように、現在のソフトウェア開発の諸問題は、コンピュータのこの処理順序探索不能性から来ているのである。

本願第１発明の原理を一般的に述べれば次の様になる。

ソフトウェアを開発したいときは必ず、ある求めたいデータ項目が存在する。本願第１発明に係る要件定義法は、一般的に、まずこの目的のデータ項目を決め、次にそのデータ項目の規定をし、ついでそのデータ生成式やデータ生成式実行条件の中に新たに出て来た新しいデータ項目を更に規定する。これを連鎖的に次々に行う。最終的に新たなデータ項目の規定がインプットデータ項目となれば完了する。夫々のデータ項目の規定は、データ生成式、データ生成式実行条件、インプット／アウトプットの属性区分、そのデータ項目の存在する記録媒体の夫々の規定によって行う。図２（連鎖式要件定義法）にデータ項目間の関係の例を示す。

即ち、図２は、データ項目間の関係を模式的に示すと共に、連鎖式データ項目規定による要件定義法（「Data Chain Requirement Definition (DCRD)法」とも呼ぶ。）を概念的に説明するための概念図である。この図はデータ項目間の関係を示すものであって、データ処理の順序を示すデータフローチャートとは全く異なることに注意されたい。

同図に示す例にあっては、目的データ項目Ａ（図中の１）は、中間データ項目Ｂ（図中の１１）、中間データ項目Ｃ（図中の１２）及び中間データ項目Ｄ（図中の１３）により規定されることを示している。さらにその中間データ項目Ｂ（図中の１１）は、Ｅファイル１１１中に格納されるインプットデータ項目Ｅ及びＦファイル１１２中に格納されるインプットデータ項目Ｆにより規定されることが示される。同様に、中間データ項目Ｃ（図中の１２）はＧ画面１２１上に存在するインプットデータ項目Ｇ及びＨ画面１２２上に存在するインプットデータ項目Ｈにより規定され、一方中間データ項目Ｄ（図中の１３）はＨ画面１２２上に存在するインプットデータ項目Ｈ及びＩファイル１３１中に格納されるイ

ンプットデータ項目Ｉにより規定されることが、それぞれ示されている。このようなデータ項目間の関係は、通常、種々異なる被開発ソフトウェアの種類、機能、コンピュータ構成等に応じて変化する。Ｅファイル１１１、Ｆファイル１１２、Ｉファイル１３１及びＧ画面１２１並びにＨ画面１２２は事実上オプションであり、また、ここには図示されないこれら以外の媒体上にインプットデータ項目が登載されていてもよい。

上記のような図２の状況下において、ここで具体的に、データ項目間の関係の中で次のような条件分岐があるとする。即ち、

もし　Ｆ＞Ｇなら、　　Ａ＝Ｂ×Ｃ／Ｄ

そうでなければ、　　Ａ＝Ｃ＋Ｇ＋Ｈ

因みに上記を従来法の一般的プログラム言語で書けば、

if　Ｆ＞Ｇ　then　Ａ＝Ｂ×Ｃ／Ｄ

else　　Ａ＝Ｃ＋Ｇ＋Ｈ

図２のようなデータ項目間の関係と上記の条件分岐がある場合、連鎖式データ項目規定による要件定義法(DCRD 法)は、次のようになる。即ち、

Ａ　＝　Ｂ×Ｃ／Ｄ　（データ生成式の規定）

Ｆ＞Ｇ　（データ生成式実行条件の規定）

アウトプットデータ。（インプット／アウトプットの属性区分の規定）

Ａ画面にある。（存在する記録媒体の規定）

（ここでの新しいデータ項目はＢ、Ｃ、Ｄ、Ｆ、Ｇだから夫々下記に規定する。以下新しいデータ項目が出て来る毎に規定を行う。）

Ａ＝Ｃ＋Ｇ＋Ｈ

Ｆ＜＝Ｇ

アウトプットデータ。

A画面にある。

B ＝ E＋F

C ＝ G／H

D ＝ H－I

E ＝ インプットデータ。Eファイルにある。

F ＝ インプットデータ。Fファイルにある。

G ＝ インプットデータ。G画面にある。

H ＝ インプットデータ。H画面にある。

I ＝ インプットデータ。Iファイルにある。

実際の作業においては、例えば、下記の表１に示すような要件定義のための単語規定作成表を用いて、新たに現れる各単語に対して規定を記述するようにすれば、漏れなく所望の効果を達成することができる。

（表１）

連鎖式要件定義(Data Chain Requirement Definition)のための単語規定作成表

データ項目番号	データ項目呼称	データ項目ID	生成式	成立条件	I	O	記録媒体呼称	記録媒体ID	備考

表１はこのようなデータ項目作成表の一である連鎖式データ項目規定

による要件定義(Data Chain Requirement Definition)のためのデータ項目作成表であるが、これに関する説明は後述する。

以上の通り、本願第１発明によれば、ソフトウェアを生産するにおいて、要件定義はまず目的のデータ項目を決め、ついでそのデータ生成式やデータ生成式実行条件の中に新たに出て来た新しいデータ項目を更に規定するという方法により、人間がデータ項目の処理順序等を考察することなく連鎖的にそこから次々と掘り起こすようにして摘出してしまう。換言すれば本願第１発明は、ソフトウェア開発において、要件定義をデータ項目の規定だけで行い、かつデータ項目の摘出を連鎖的機械的に簡単に行う方法である。

このようにデータ項目によって要件定義を行うためのソフトウェアツールが準備されている。画面上に現れる表１のデータ項目の呼称欄に目的であるアウトプットのデータ項目の呼称、ＩＤ（識別符号）、データ生成式(式中のデータ項目は呼称で書く)、データ生成式実行条件、インプット(Ｉ)／アウトプット（Ｏ）の属性区分、そのデータ項目が存在する記録媒体の呼称とＩＤをインプットすれば、新しい行に新しくデータ生成式やデータ生成式実行条件に現れたデータ項目の呼称と番号が自動的に現れる。新しい行に次々に新しく現れるそれらのデータ項目について一つ一つ規定して機械的に欄を埋めて行けばよい。最終的に新しく現れたデータ項目がインプットデータになれば要件定義は終了する。

このようにデータ項目の規定によって完了した要件定義は、そのデータ項目の並び方がデータ項目の処理順序と全く関係が無い。従来、データ項目の処理順序は、データ項目を並べている順序或いは条件分岐や反復の指示等で人がプログラムによって指示してやらなければならなかった。本願第１発明によって行われるデータ項目による連鎖的機械的要件定義も、従来法のプログラムではその後人間がデータ項目の処理順序を

分析探索してそれをプログラムによって指示しなければならない。

そこで本願に係る第２の発明（以下、「本願第２発明」ということもある。）は、人がプログラムによってデータ項目の処理順序を指示するのではなく、本願第１発明によって機械的に行われた要件定義をもとに、データ項目の処理順序を自動的に決定する「プログラムの自動的開発手法」（例えば上記の「Ｌｙｅｅ（登録商標）」で規定される手法）とを組み合わせることによって、結局最終製品たるソフトウェアの開発自体を自動的に行うものである。

つまり本願第２発明によれば、本願第１発明を基礎とした上で、機械的に得られた要件定義をもとに、データ項目の処理順序を自動的に見出すか、或いはデータを自動的に正しい順序で成立させて行くプログラム自動的開発手法、例えばＬｙｅｅ（登録商標）、を用いるとするので、所望のソフトウェア開発が終局に至るまで自動化されるという劇的な効果がもたらされるものである。本願第１発明については上記に詳述し、またＬｙｅｅ（登録商標）発明については上記した文献に詳細に開示したので、ここではそれを指摘するにとどめ同じ説明の繰り返しは省略する。

以上述べたことを要すれば、本願第１発明によれば、ソフトウェアを生産するにおいて、まず目的のデータ項目を決め、ついでそのデータ生成式やデータ生成式実行条件の中に新たに出て来た新しいデータ項目を更に規定することにより、人間がデータ項目の処理順序等を考察することなく、データ項目の摘出及び要件定義を連鎖的機械的に簡単に行うことが可能となる。

また、本願第２発明によれば、本願第１発明によって行われた機械的要件定義と、それらの規定されたデータ項目に基づいてデータ項目の処理順序を自動的に見出して、或いはデータを自動的に正しい順序で成立

させて行って、プログラムを自動的に開発する手法（例えば上記の「Ｌｙｅｅ（登録商標）」で規定される手法）とを用いるので、ソフトウェアの開発を自動的に行うことが可能となる。

（第２の実施形態）

次に、本願の第３の発明として、連鎖式要件定義および要件変更ツールについて説明する。

連鎖式要件定義法によって実際に要件単語を新規に規定する場合或いは要件の変更に応じて要件単語を変更する場合、それを実行するためのツールであるコンピュータソフトウェアを使うとよい。

ツールソフトの基本原理は極めて簡単である。ここでもう一度確認するが、要件の変更とは「要件単語の定義即ち個々の要件単語の規定の変更」でありその変更・削除・追加である。ツールソフトの基本原理は本発明の基本原理と当然同じであり下記の通りである。

（ａ）　変更する要件単語の変更前の定義の画面表示（形式は表１）とそれに対する変更操作

（ｂ）　変更する単語の変更前と変更後の第１リンク定義単語群と第１リンク包含単語群の定義の画面表示

（この（ａ）と（ｂ）が前記のＹオペレーションである。）

（ｃ）　上記（ｂ）の個々の単語について、規定変更が必要かどうかの人による検討

（ｄ）　上記（ｃ）の結果、変更の必要がある単語について上記の（ａ）、（ｂ）、（ｃ）と同様の操作

（ｅ）　上記（ｄ）の結果ある或要件単語から派生する総ての単語の規定に変更の必要がなくなれば当該要件単語の変更操作は完了する。

変更が要件単語の新規追加の場合は、新規にその定義をすることは規

定なしの状態から規定ありの状態への変更であるから、新規定義操作として（ａ）の変更操作を行えばよく、次にそれに応じて（ｂ）、（ｃ）、（ｄ）、（ｅ）を行えばよい。

全くの新規に要件を定義するために要件単語の定義をしていく場合は変更すべき要件単語がいまだ存在しない。その定義をするためには新規定義操作として何もないものからの上記（ａ）の変更操作を行えばよく、次にそれに応じた第１リンク定義単語群の定義を（ｂ）、（ｃ）、（ｄ）、（ｅ）で新規定義操作を行えばよい。

以上から、ツールソフトとしては変更のためのものも新規開発のためのものも全く同じものである。元々新規開発の場合も、初めから要件定義が全部確定している訳ではなく個々の要件定義を試行錯誤的に変更を繰り返しながら行っていく間に全体の要件定義が構築されるのが現実である。したがってその途中の変更は本来避けられない。その観点から新規開発向けも変更向けも同じツールであるべきである。そしてその頻繁な変更が容易に行えるならばソフトウェア開発としては大変望ましい。その能力を持つ連鎖式要件定義法のツールソフトはソフトウェア開発方法としてその生産効率を飛躍的に高める。

本発明の多くの効果および利点は明細書の詳細な説明から明白である。これまでに説明してきた効果及び利点を要約すると次のようになる。

上記「従来の技術」の項で述べたように、プログラムの変更は常に必要である。既に完成されたプログラムの場合でも、業務環境や経営の方針の変化に応じて要件は常に変化しているので、それに応ずるプログラムの変更は常に必要である。

また新規に開発するプログラムの場合でも、初めに業務システムやプログラムシステムが総て決まっていることはむしろ少ないので、プログラムはプログラミング過程で変更しながら作成して行かなければならな

い。

一方、上記「従来の技術」の項で述べたように、従来法のプログラムではその命令文の順序にしたがって計算が実行される。したがって処理順序は人が命令文を書く順序によって或いは条件付分岐の行く先を書くことによって指示しているし指示しなければならない。そうすると処理順序が決められているので１つのプログラムの中の１つのデータ項目の値はそれが処理されるまでの前処理と当該の処理によって決まる。したがって１つのデータ項目の値は１つのプログラムの中の何処に書かれているかによる。したがって前処理を理解してさえいれば１つのデータ項目は１つのプログラムの中で別のデータ生成式を持ってもよいし別の値を持ってもよい。つまり１つのデータ項目が１つのプログラムの中でどんな前処理をされたかに拘わらず同じ値を持つということは保証されていないのである。結局、処理順序を人が決めてやらなければならない代わりにデータ項目の定義や値に自由度が出てきてしまうのである。だから１つのデータ項目は１つのプログラムの中で何処でどんな定義で使われているのかは作った人以外にはわからない。作った人でも長くは覚えてはいない。

このような状況において、要件変更に応じてプログラムを変更するとき、１つのプログラムの変更がプログラムの他の部分に与える影響を考え必要ならばそれらを正しく変更しなければならない。そのために従来法では上記のようにデータ項目の定義や値に自由度が出てくるためにプログラムは人が法則もなく勝手に作ることができるから、必要な変更がプログラム全体の何処にあるか分らなくなってしまうのである。必要な変更の総てを見つけて正しく変更することは容易ではない。１箇所変えればそのことによるプログラム全体への影響は機械的には分らないからである。総てを正しく変更するためには、人がプログラムの中の総ての

命令文を詳細にたどって変更すべきところを摘出して変更することしかないのである。

本発明の連鎖式要件定義方法は、要件の定義においても要件の変更においても従来法の上記の問題を総て解消してしまうのである。

さらにこの連鎖式要件定義法によって定義された要件単語群を、処理順序を問わない自動プログラミング法（例えばＬｙｅｅ（登録商標）。特許文献１乃至６参照）に適用すれば、ソフトウェア開発が新規開発においても要件の変更においても、要件の定義過程の端緒からプログラミングに至るまで人の考えなければならないことを極小化して自動化され、劇的な効果を得ることができる。

本発明は当該技術分野における通常の知識を有する者にとって修正および改変が容易に数多くなし得るので、図示および記述されたものと寸分違わぬ構成および動作に本発明を限定することは望ましくないことであり、従って、あらゆる適切な改変体および等価体は本発明の範囲に含まれるものと見なされうる。前述の本発明に係る実施の具体的形態の説明および例示によって詳細に記述されたが、本願の特許請求の範囲のみならず本発明に係る開示事項全体に定義された本発明の範囲から逸脱することなしに、修正、置換、および、変更が数多く可能である。

また、本願に係る発明は、その適用において、上記の記述において説明されるか、或いは、図面に示された要素の詳細な解釈及び組み合わせに限定されるものではない。本発明は、他の実施形態が可能であり、種々の方法で実用および実施可能である。また、ここで用いられた語法および用語は記述を目的とするものであり、限定的に働くものとみなされてはならない。

従って、当該技術分野における通常の知識を有する者は、本開示の基調となる概念は、本発明の幾つかの目的を実施するための他の構造、方

法、及び、システムを設計するための基礎として容易に利用され得ることを理解するはずである。従って、本発明の趣旨および範囲から逸脱しない限り、本願の特許請求の範囲にはそのような等価な解釈が含まれるものと見なされるものである。

また、上記ではコンピュータソフトウェアの自動開発法について主として説明したが、本発明に係る技術思想は、例えばコンピュータソフトウェアの自動開発装置、自動開発プログラム、自動開発プログラムを記録した記録媒体、伝送媒体、紙媒体としても、また、自動開発プログラムを登載したコンピュータ・装置、自動開発プログラムを実行するクライアント・サーバ形式等といったカテゴリーにおいても実現、利用可能であることはいうまでもない。

さらに本発明は、単一プロセッサ、単一ハードディスクドライブ、及び、単一ローカルメモリを備えたコンピュータシステムに限らず、当該システムのオプションとして、任意の複数または組み合わせプロセッサ又は記憶デバイスを装備するにも適している。コンピュータシステムは、精巧な計算器、掌タイプコンピュータ、ラップトップ／ノートブックコンピュータ、ミニコンピュータ、メインフレームコンピュータ、及び、スーパーコンピュータ、ならびに、これらの処理システムネットワーク組み合わせを含む。本発明の原理に従って作動する任意の適切な処理システムによって代替されうるし、また、これらと組み合わせて用いることも可能である。

また、本発明に係る技術思想は、もとよりあらゆる種類のプログラミング言語に対応可能である。さらに本発明に係る技術思想は、あらゆる種類・機能のアプリケーションソフトウェアに対しても適用可能である。

また、本願第２発明については、本願第１発明と組み合わせる対象は、本願第１発明によって得られた機械的要件定義によってプログラムを自

動的に開発する手法であればよく、上述の「Ｌｙｅｅ（登録商標）」にその対象が限定されるわけではない。

またさらに本願発明は、その技術思想の同一及び等価に及ぶ範囲において様々な変形、追加、置換、拡大、縮小等を許容するものである。また、本願発明を用いて生産されるソフトウェアが、その２次的生産品に登載されて商品化された場合であっても、本願発明の価値は何ら減ずるものではない。

上述したように本発明の要件定義方法は、データ項目の摘出及び要件定義を連鎖的機械的に簡単に行うことが可能とする。

また、本発明のコンピュータソフトウェアの自動開発方法は、コンピュータソフトウェアの開発工程から人間が必ず指示しなければならないもの以外をなくし、最終的にソフトウェアの開発を極力機械的にし、自動化を可能にする。

また、本発明の連鎖式要件定義方法によれば、要件の新規定義においても要件の変更においても従来法の「１つのプログラムの変更の影響によって必要となる要変更箇所がプログラム全体の何処にあるか分らなくなってしまう」という問題を総て解消してしまうのである。

さらにこの連鎖式要件定義法によって定義された要件単語群を、処理順序を問わない自動プログラミング法（例えばＬｙｅｅ（登録商標））に適用すれば、ソフトウェア開発が新規開発においても要件の変更においても、要件の定義過程の端緒からプログラミングに至るまで人の考えなければならないことを極小化して自動化され、劇的な効果を得ることができる。

産業上の利用可能性

本発明では、データ項目の摘出及び要件定義を連鎖的機械的に簡単に

行うことを可能とし、また、コンピュータソフトウェアの開発工程から人間が必ず指示しなければならないもの以外をなくし、最終的にソフトウェアの開発を極力機械的にし、自動化を可能にすることで、ソフトウェア生産の大幅な効率向上、生産性向上、品質向上等、ソフトウェア産業上に大きな効果をもたらす。

請　求　の　範　囲

1．（ａ）開発対象のコンピュータソフトウェアが最終的に求めようとするアウトプットデータ項目を総て取り出すステップと、

（ｂ）該取り出されたアウトプットデータ項目の一をデータ生成式及びそのデータ生成式実行条件により規定するステップと、

（ｃ）前記データ生成式及びそのデータ生成式実行条件を規定するために現れた新たなデータ項目の総てに対し、当該総ての新たなデータ項目のそれぞれを別のデータ生成式及びそのデータ生成式実行条件により規定するステップと、

（ｄ）前記ステップ（ｃ）を、当該データ生成式を構成するのがインプットデータ項目のみとなるまで繰り返すステップと、

（ｅ）前記ステップ（ａ）乃至ステップ（ｄ）を、前記最終的に求めようとするアウトプットデータ項目の総てについて実行し、この実行の結果得られたデータ生成式及びそのデータ生成式実行条件を要件定義とするステップと

を備えることを特徴とする要件定義方法。

2．前記ステップ（ｂ）またはステップ（ｃ）におけるデータ項目は、当該データ生成式及びそのデータ生成式実行条件に加えて、さらに、インプット／アウトプットの属性区分及びそのデータ項目の存在する記録媒体の明示によっても規定され、

前記ステップ（ｅ）において、前記実行の結果得られたデータ生成式及びそのデータ生成式実行条件に加え、さらに、インプット／アウトプットの属性区分及びそのデータ項目の存在する記録媒体に係る規定をも要件定義とする

ことを特徴とする請求項１記載の要件定義方法。

3．（ａ）前記請求項１もしくは２記載の要件定義方法を用いて要件定義

を得るステップと、

（ｂ）前記得られた要件定義中に規定されたデータ項目に基づいて、データ項目の処理順序を自動的に見出すことで或いはデータを自動的に正しい順序で成立させて行くことでプログラムを自動的に開発する手法に適用し、該適用の結果所望のソフトウェアを得るステップと

を備えることを特徴とするコンピュータソフトウェアの自動開発方法。

４．ソフトウェアの要件として規定される要件単語（＝データ項目）群に係る、

該単語の名前、

該単語に対応する値を得るデータ生成式（インプットによって値が得られることを含む。）、

該単語に対応する値が成立するための条件（データ生成式実行条件）、

該単語がインプットかアウトプットかの属性区分、

該単語の存在する記録媒体

で規定される情報を、規定された要件単語の配列の順序を問わないで自動プログラミングができる手法に適用して自動的に作成されるプログラムの要件の変更にあたり、

（ａ）変更すべき要件単語自身の規定を変更（変更は削除、追加を含む。）する操作と、

（ｂ）前記（ａ）の当該単語の規定の変更の影響によって規定変更が必要となる可能性がある要件単語として、当該要件単語の変更前と変更後の第１リンク定義単語と第１リンク包含単語とを摘出する操作と、

（ｃ）前記摘出された個々の単語について、その規定の変更が必要かどうかを検討する操作と、

（ｄ）変更の必要がある単語について前記（ａ）乃至（ｃ）の操作を

繰り返す操作と

を備えることを特徴とする要件単語の変更方法。

5．ソフトウェアの要件として規定される要件単語（＝データ項目）群に係る、

該単語の名前、

該単語に対応する値を得るデータ生成式（インプットによって値が得られることを含む。）、

該単語に対応する値が成立するための条件（データ生成式実行条件）、

該単語がインプットかアウトプットかの属性区分、

該単語の存在する記録媒体

で規定される情報を、規定された要件単語の配列の順序を問わないで自動プログラミングができる手法に適用して自動的に作成されるプログラムの新規開発においては、新規要件規定作業とその修正作業との両方が必要であるので、プログラムの新規開発にあたり、新規の要件規定は何もない状態からの変更と考えて、

（ａ）変更すべき要件単語自身の規定を変更（変更は削除、追加を含む。）する操作と、

（ｂ）前記（ａ）の当該単語の規定の変更の影響によって規定変更が必要となる可能性がある要件単語として、当該要件単語の変更前と変更後の第１リンク定義単語と第１リンク包含単語とを摘出する操作と、

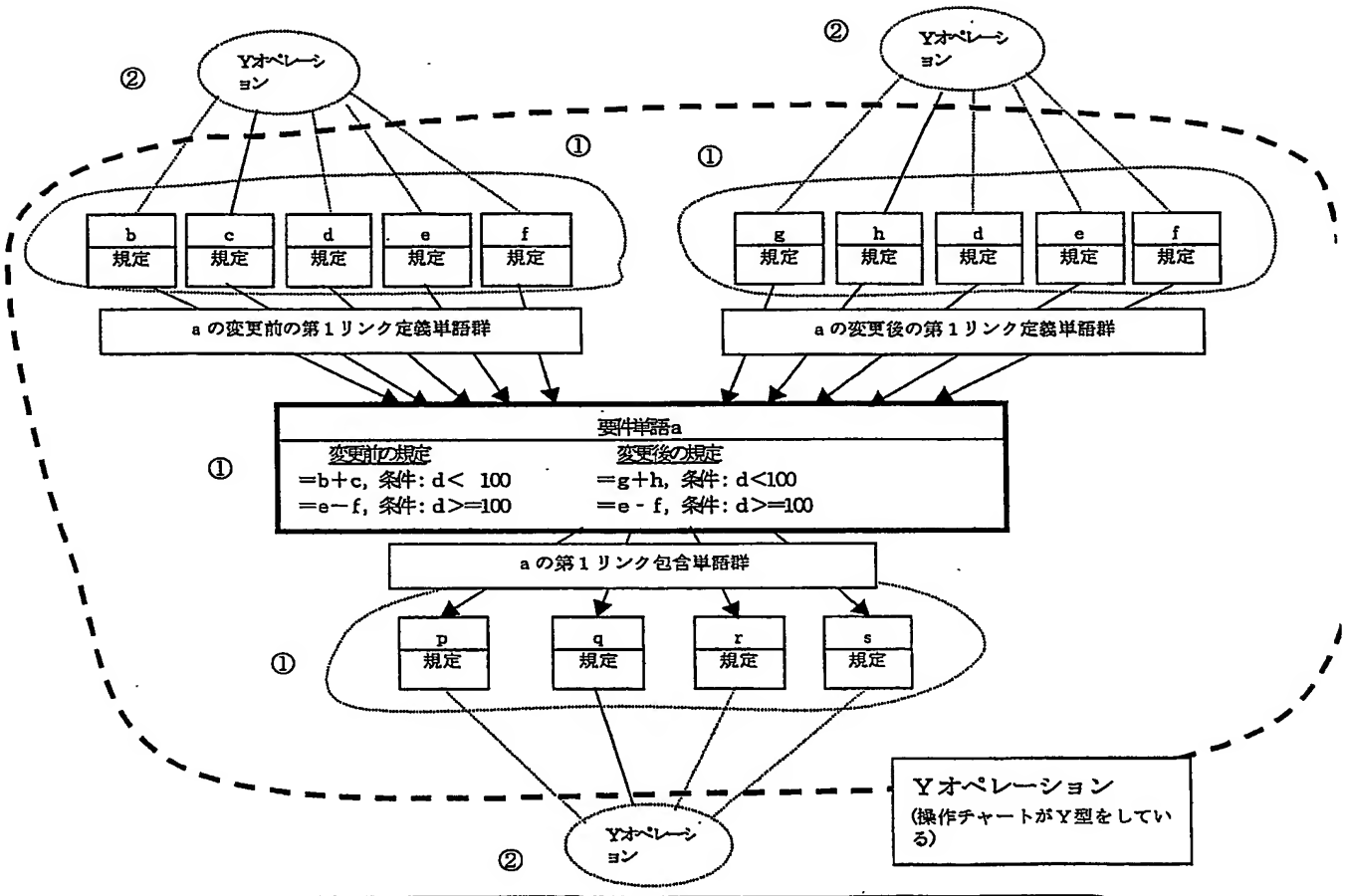
（ｃ）前記摘出された個々の単語について、その規定の変更が必要かどうかを検討する操作と、

（ｄ）変更の必要がある単語について前記（ａ）乃至（ｃ）の操作を繰り返す操作と

を備えることを特徴とする要件単語の新規規定方法。

第１図

要件単語の規定変更の手順図（Ｙオペレーション）

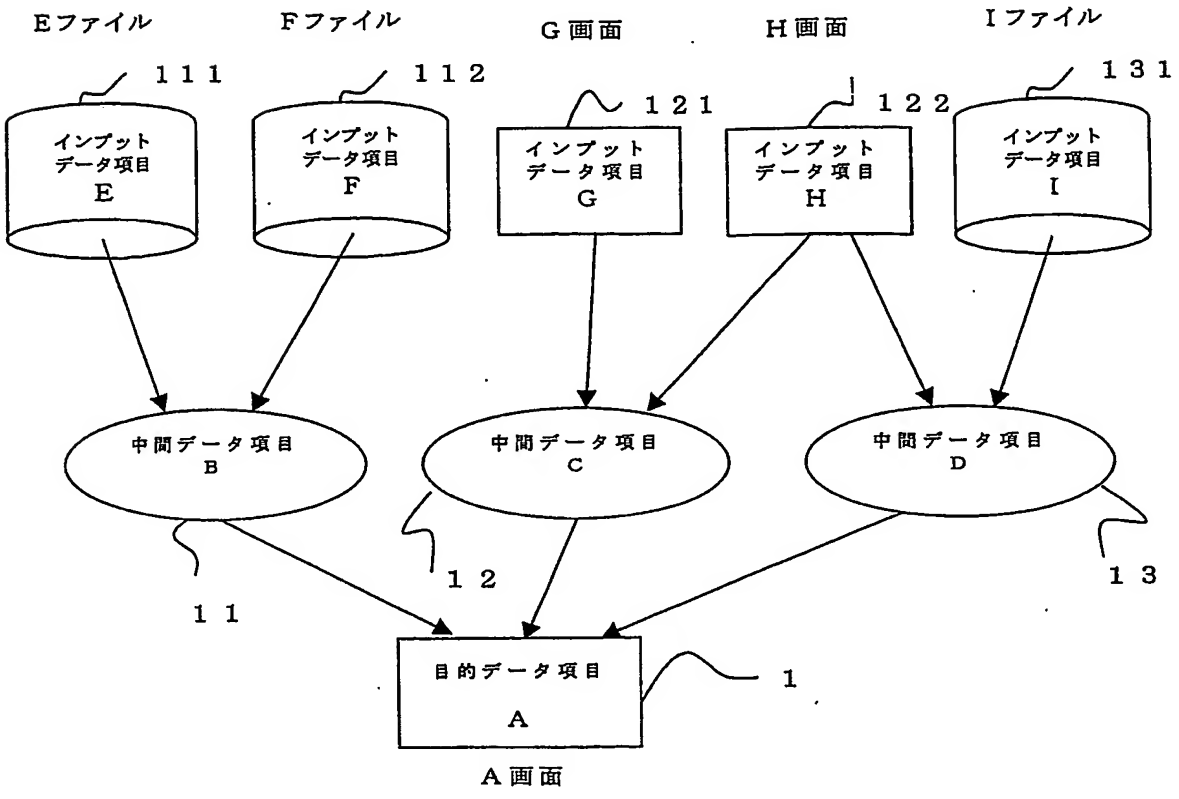


① この要件単語ａの規定を変更する。
変更前と変更後の第１リンク定義単語群を摘出する。
（ｂ，ｃ，ｄ，ｅ，ｆ，　と　ｇ，ｈ，ｄ，ｅ，ｆ）
この要件単語の第１リンク包含単語群を摘出する。
（ｐ，ｑ，ｒ，ｓ）

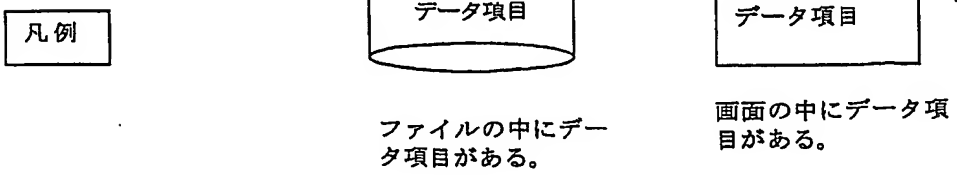
② 第１リンク定義単語群と第１リンク包含単語群の個々の要件単語の規定に変更が必要かどうか検討する。
変更が１つでもあれば、その当該個別単語についてＹオペレーションを行う。
全てのリンク単語に変更がなくなるまで続ける。一つもなければそれ以降のリンク単語に変更は波及しないので終了する。

第２図

連鎖式要件定義法
(Data Chain Requirement Definition (DCRD)法)
データ項目間の関係図



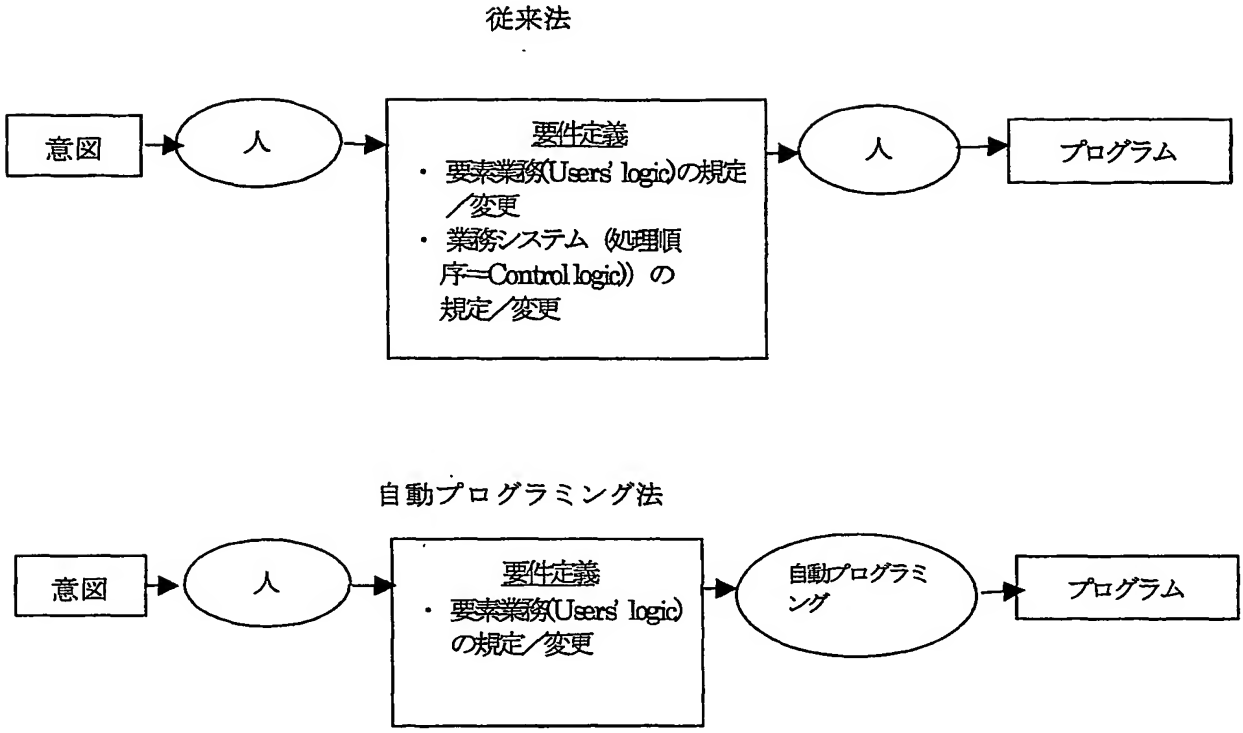
上図のデータ項目Ａ，Ｂ，Ｃ，Ｄの関係は、Ａというデータ項目はＢデータ項目，Ｃデータ項目，Ｄデータ項目から成ることを示す。



ファイルの中にデータ項目がある。

画面の中にデータ項目がある。

第３図


従来法
意図
人
要件定義
・要素業務(Users' logic)の規定／変更
・業務システム（処理順序=Control logic)）の規定／変更
人
プログラム
自動プログラミング法
意図
人
要件定義
・要素業務(Users' logic)の規定／変更
自動プログラミング
プログラム

INTERNATIONAL SEARCH REPORT

International application No.
PCT/JP03/11486

A. CLASSIFICATION OF SUBJECT MATTER
Int.Cl[7] G06F9/44

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)
Int.Cl[7] G06F9/44

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched
Jitsuyo Shinan Koho 1926-1996 Jitsuyo Shinan Toroku Koho 1996-2003
Kokai Jitsuyo Shinan Koho 1971-2003 Toroku Jitsuyo Shinan Koho 1994-2003

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X Y	JP 6-139224 A (Hitachi, Ltd.), 20 May, 1994 (20.05.94), Full text; all drawings (Family: none)	1,2 3
Y	JP 2001-5651 A (The Institute of Computer Based Software Methodology and Technology), 12 January, 2001 (12.01.01), Full text; all drawings & WO 00/79385 A1 & EP 1244006 A1 & CA 2414110 A & AU 5427900 A & CN 1376279 T	3-5
Y	JP 9-106343 A (Hitachi, Ltd., Hitachi Software Engineering Co., Ltd.), 12 April, 1997 (12.04.97), Full text; all drawings (Family: none)	4,5

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

* Special categories of cited documents:
"A" document defining the general state of the art which is not considered to be of particular relevance
"E" earlier document but published on or after the international filing date
"L" document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)
"O" document referring to an oral disclosure, use, exhibition or other means
"P" document published prior to the international filing date but later than the priority date claimed
"T" later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"X" document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"Y" document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"&" document member of the same patent family

Date of the actual completion of the international search	Date of mailing of the international search report
17 December, 2003 (17.12.03)	13 January, 2004 (13.01.04)
Name and mailing address of the ISA/ Japanese Patent Office Facsimile No.	Authorized officer Telephone No.

Form PCT/ISA/210 (second sheet) (July 1998)

国際調査報告　　国際出願番号　PCT／JP03／11486

A.　発明の属する分野の分類（国際特許分類（IPC））

Int Cl⁷　G06F9／44

B.　調査を行った分野

調査を行った最小限資料（国際特許分類（IPC））

Int Cl⁷　G06F9／44

最小限資料以外の資料で調査を行った分野に含まれるもの
日本国実用新案公報　1926年－1996年
日本国公開実用新案公報　1971年－2003年
日本国実用新案登録公報　1996年－2003年
日本国登録実用新案公報　1994年－2003年

国際調査で使用した電子データベース（データベースの名称、調査に使用した用語）

C.　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求の範囲の番号
X Y	JP　6－139224　A（株式会社日立製作所） 1994．05．20，全文，全図（ファミリーなし）	1，2 3
Y	JP　2001－5651　A（ソフトウェア生産技術研究所株式会社，株式会社アイエスデー研究所）2001．01．12，全文，全図　＆　WO　00／79385　A1　＆　EP　1244006　A1　＆　CA　2414110　A　＆　AU　5427900　A　＆　CN　1376279　T	3－5

☒ C欄の続きにも文献が列挙されている。　☐ パテントファミリーに関する別紙を参照。

＊　引用文献のカテゴリー
「A」特に関連のある文献ではなく、一般的技術水準を示すもの
「E」国際出願日前の出願または特許であるが、国際出願日以後に公表されたもの
「L」優先権主張に疑義を提起する文献又は他の文献の発行日若しくは他の特別な理由を確立するために引用する文献（理由を付す）
「O」口頭による開示、使用、展示等に言及する文献
「P」国際出願日前で、かつ優先権の主張の基礎となる出願の日の後に公表された文献
「T」国際出願日又は優先日後に公表された文献であって出願と矛盾するものではなく、発明の原理又は理論の理解のために引用するもの
「X」特に関連のある文献であって、当該文献のみで発明の新規性又は進歩性がないと考えられるもの
「Y」特に関連のある文献であって、当該文献と他の1以上の文献との、当業者にとって自明である組合せによって進歩性がないと考えられるもの
「&」同一パテントファミリー文献

国際調査を完了した日	国際調査報告の発送日
17．12．03	13.01.04

国際調査機関の名称及びあて先	特許庁審査官（権限のある職員）	5B	9366
日本国特許庁（ISA／JP） 郵便番号100－8915 東京都千代田区霞が関三丁目4番3号	漆原　孝治 電話番号　03－3581－1101　内線　3546		

様式PCT／ISA／210（第2ページ）（1998年7月）

Ｃ（続き）．　関連すると認められる文献

引用文献のカテゴリー＊	引用文献名　及び一部の箇所が関連するときは、その関連する箇所の表示	関連する請求の範囲の番号
Ｙ	ＪＰ　９－１０６３４３　Ａ（株式会社日立製作所，日立ソフトウエアエンジニアリング株式会社）１９９７．０４．１２，全文，全図　（ファミリーなし））	４，５